CROSS-LANGUAGE TEXT CLASSIFICATION WITH CONVOLUTIONAL NEURAL NETWORKS FROM SCRATCH

Musbah Zaid Enweiji
National University of Kyiv-Taras Shevchenko
64/13 Volodymyrska str., Kyiv, Ukraine, 01601
s.cc85@yahoo.com

Taras Lehinevych
National University of "Kyiv-Mohyla Academy"
2 Skovorody str., Kyiv, Ukraine, 04655
taras.lehinevych@gmail.com

Andrey Glybovets
National University of "Kyiv-Mohyla Academy"
2 Skovorody str., Kyiv, Ukraine, 04655
andriy@glybovets.com.ua

Abstract

Cross language classification is an important task in multilingual learning, where documents in different languages often share the same set of categories. The main goal is to reduce the labeling cost of training classification model for each individual language. The novel approach by using Convolutional Neural Networks for multilingual language classification is proposed in this article. It learns representation of knowledge gained from languages. Moreover, current method works for new individual language, which was not used in training. The results of empirical study on large dataset of 21 languages demonstrate robustness and competitiveness of the presented approach.

Keywords: text classification, convolutional neural network, cross-language text classification, multilingual classification, transfer learning, inductive transfer, supervised learning, artificial neural network.

DOI: 10.21303/2461-4262.2017.00304

1. Introduction

Text classification or text categorization problem is currently one of the most observed in the field of information and computer sciences. The task is to assign a text to one or more classes or categories. Classification is a popular technique and has been applied to spam filtering, email routing, sentiment analysis and etc. This may be done "manually" or algorithmically. Machine learning is an outstanding approach in automated text classification [1, 2] and categorization [3]. Moreover, text clustering or unsupervised text classification has been used to enhance the information retrieval. This is based on the clustering hypothesis, which states that text having similar contents is also relevant to the same query [4]. A fixed collection of text is clustered into groups or clusters that have similar contents. The similarity between texts is usually measured with the associative coefficients from the vector space model, e. g., the cosine coefficient.

There are numerous automatic text classification techniques that include:

− Expectation maximization (EM);
− Naive Bayes classifier;
− Tf-idf;
− Latent Semantic Indexing;
− Latent Dirichlet Allocation;
− Support Vector Machines (SVM);
− K-nearest neighbor algorithms;
− Decision Trees such as ID3 or C4.5.

Besides these techniques, there are also neural networks and deep learning approaches that were responsible for the boost in Machine Learning [5]. The work is focused on Convolutional Neural Networks (CNNs). It is commonly assumed that CNN is only used in Computer Vision [6]

and it is particularly true, because convolutional neural networks are major breakthroughs in image classification tasks and now are the most successful approach in most Computer Vision Systems.

CNNs are made up of neurons that have learnable weights and biases as ordinary neural networks. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores on the other. Moreover, they still have a loss function (e. g. SVM/Softmax) on the last fully-connected layer and all the tips and tricks that were developed for learning regular neural networks still apply.

Recently, CNNs are applied to problems in Natural Language Processing (NLP) and the obtained results are promising and interesting for the future research.

Most machine learning approaches on text understanding and classification consist in tokenizing a string of characters into structures such as words, phrases, sentences, and paragraphs. Usually, some statistical classification algorithm is applied to the statistics of these structures [7]. These techniques work well enough for a narrowly defined domain. However, the prior knowledge is required: a pre-defined dictionary of interested words, a parser to handle variations such as word morphological changes and specialization to a particular language.

With the advancement of deep learning and the availability of a large dataset, methods of handling text understanding using deep learning techniques have gradually become available. One technique that draws great interests is word2vec [8]. Inspired by traditional language models, this technique constructs a representation of words into a vector of fixed length trained under a large corpus. Numerous of techniques try to apply these representation or similar approaches with an engineered language model and have good results for various tasks [2]. Moreover, the neural networks will perform multilingual classification. The term "multilingual" came from the information retrieval and it also has a specific meaning in cross-language information retrieval, where a text collection is multilingual. This article shows that cross-language multi-label text classification can be handled by a deep learning system without artificially embedding knowledge about words, phrases, sentences or any other syntactic or semantic structures associated with a language. In the following sections, firstly, related work is reviewed, and then introduce the multilingual dataset. Materials and methods section describes neural network architecture and model description for cross-language classification. Experiments and evaluations are presented in sections 4 and 5. Finally, the conclusions are done in section 6.

For the first time the problem of cross-language text classification was explicitly considered in cross-lingual text categorization paper [9], which is predated by work in cross-language information retrieval (CLIR), where similar problems are addressed [10].

Traditional approaches to cross-language text classification or/and CLIR are connected with the usage of linguistic resources such as bilingual dictionaries or parallel corpora to induce correspondences between two languages [11, 12], for example, method [13] which performs latent semantic analysis on a parallel corpus and is considered as seminal work in CLIR. Later on it was improved by Li and Taylor [14] by employing kernel canonical correlation analysis instead of latent semantic analysis. The limitations of these approaches are: dependence on parallel corpus and computational complexity.

The use of automatic machine translation approaches was studied in cross-language text clarification [12, 15]. These methods typically translate the text into the source or target language. Usually, the second step is to reduce the noise introduced by the machine translation using dimensionality reduction [16] or semi-supervised learning [15]. Obviously, the performance of such classifiers depends on the quality of the automatic machine translation. The paper introduces an approach for cross-language multi-label text classification by using deep convolutional neural network. Our extensive empirical study on a large dataset of multilingual texts suggests that the proposed solution has multiple benefits. First of all, it does not require parallel data between all languages or bilingual dictionaries, what is often a bottleneck, because in many real world scenarios such parallel data may not be available. Secondly, there is no need to use artificially embedding knowledge about words, phrases, sentences or any other syntactic or semantic structures associated

with a language. Finally, representation of text for any language is learned by neural network and could be used in other application.

2. Materials and Methods

In order to train and test CNN for cross-language multi-label classification proper data is required. The dataset was created based on a text extracted from Wikipedia articles, for this purpose, the crawler that iterates over DBpedia (public dataset of Wikipedia) was implemented. The dataset consists of article URLs, categories and their confidence. The texts of the articles were collected by crawling of the corresponding URLs. The dataset of 23 languages with defined multiple categories per every article (**Table 1**) was constructed.

The numbers of articles in Portuguese and Swedish are bigger than, for example, Bulgarian. Besides, every category has a different number of articles and it is assigned to across different languages. Moreover, every article has a different size, the smallest ones could have two sentences, the largest approximately ten pages of text. That is why, in order to have a "balance" dataset, the length of each article equal to 1.500 words was fixed, the longer text was simply cut, the shorter was padded (special token was appending). The number of articles per every category varies from 150 to 400 per language and respectively the total number of articles per language after pre-processing is in range: 39.000−78.000. It was filtered out the exceptional categories that are assigned for less than 150 articles across the whole dataset (**Fig. 1**). To this end, the dataset contains almost 450.000 articles and more than 300 categories. In this article, all the results are presented based on collected data.

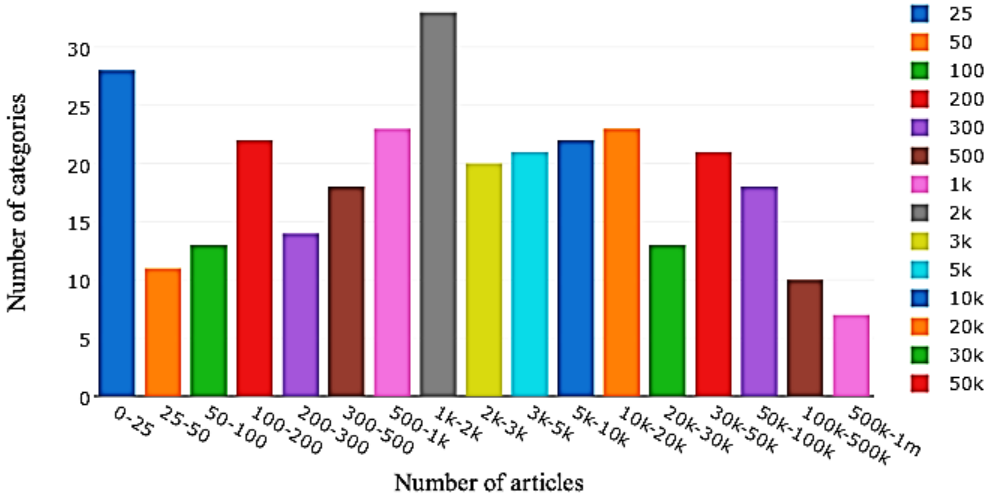

Fig. 1. Number of categories that are represented by number of articles (ranges) across all languages

The dataset does not come with an official train/test split, so it was simply divided into train, validation and test sets, 70 %, 15 % and 15 % accordingly. Data pre-processing steps are:

− load sentences from the raw data files;

− clean the text data using regular expression in order to match only words;

− append special token to each article text with the length less than 1.500 words;

− this step is required, because each example in a batch must be of the same length;

− build a vocabulary index and map each word to an integer between 0 and the vocabulary size. Each article text becomes a vector of integers;

− build a category index and map each category to an integer between 0 and the number of categories. For each article text, a vector of integers was gotten.

The last two steps allow to represent the text as a list of numbers on the word level. However, the vocabulary size grows with every additional language and consists of all words from training set. To solve this problem word embedding technique is used, it was originally introduced

by Bengio more than a decade ago [17]. A word embedding $W : words \to R^n$ is a parameterized function mapping words from vocabulary to high-dimensional vectors. The embedding layer is the first stage in this model: input is a list of numbers (each number is index of word in the vocabulary), output is 512-dimensional vector. The mapping function is learned by training procedure in order to have meaningful vectors for multilingual classification task. Let assume $x_i \in R^k$ is the k-dimensional word vector corresponding to the i-th word in the text of article.

$$x_{1:n} = x_1 * x_2 * ... * x_n .$$

Then an article will be the concatenation of x_i, where n is the number of words in an article.

Applying convolutional networks to text classification or natural language processing at large was explored in the literature. It has been shown that CNNs can be directly applied to distributed [18, 19] or discrete [20] embedding of words, without any knowledge of the syntactic or semantic structures of a language. These approaches have been proven to be competitive with traditional models. This article explores treating the text as a kind of raw signal at the word level, and applying temporal (one-dimensional) CNN to it on purpose to assign categories. The example is presented in the **Fig. 2**.

Table 1
Describes the number of articles and categories per language

Language	Abbreviation	Articles	Categories
Belarusian	be	46 899	231
Bulgarian	bg	120 375	280
Catalan	ca	321 939	286
Czech	cs	217 920	290
Danish	da	138 876	284
Estonian	et	72 082	257
Finnish	fi	25 361	293
Croatian	hr	95 998	279
Hungarian	hu	202 290	279
Indonesian	id	126 272	279
Icelandic	is	18 107	231
Lithuanian	lt	111 712	262
Latvian	lv	32 261	255
Norwegian (Bokmål)	no	277 582	294
Portuguese	pt	673 440	305
Romanian	ro	191 744	274
Slovak	sk	155 087	276
Slovenian	sl	98 299	258
Albanian	sq	31 208	214
Serbian	sr	163 530	269
Swedish	sv	722 512	303
Turkish	tr	175 918	284
Ukrainian	uk	376 432	287

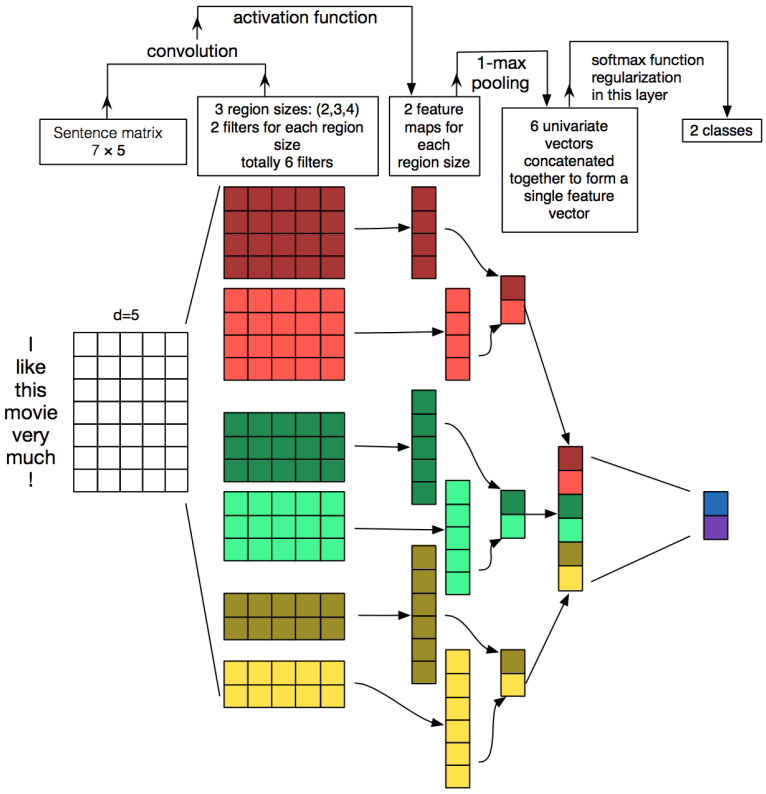

Fig. 2. Model architecture with two channels for an example sentence [19]

The model architecture is a deeper extended variant similar of Kim Yoon's CNNs for sentence classification [19] and inspired by Collobert [21], Kalchbrenne [22] and Alexis Conneau [23]. The convolution operation involves a filter that defined as $w \in R^{hk}$ The filter is applied to a window of l words in order to produce a new feature $c_i = f(w * x_{i:i+l} + b)$, where b is a bias and f is non-linear activation function. Production of the feature map requires applying this filter with a different window length of the words.

$$c = [c_1, c_2, ..., c_{n,l+1}],$$

where $c \in R^{nl+1}$.

On the next step a max pooling operation [21] is used over the feature map. As a characteristic of a particular filter maximum value $c' = \max(c)$ is chosen. The main purpose is to capture the most important characteristic for the whole feature map. This pooling scheme is common for computer vision approach and also it naturally deals with article lengths. The model performs convolution operations over $x_{\{1:n\}}$ vectors of article representation. The data is passed from embedding layer to the first convolutional layer that is defined the 32 window size filters, then to the second with 16 window size. The third layer is combination of three convolutional ones with 3, 4 and 5 window sizes 2, each layer takes as an input the output of second layer and produces separate outcomes, as showed in **Fig. 3**. The outcomes are concatenated and feed to the feedforward layer.

Moreover, it is important to mention that for all convolutional operations the stride size is 1 (i. e. the shifting length of filters at each step) and wide convolution or zero-padding is used (i. e. adding zero elements on the matrix edges for elements that don't have all neighboring elements). The ReLU is activation function that was chosen because of speed characteristics. The next component is a fully connected layer, each input neuron is connected to each output neuron in the next layer, which output is the probability distribution over categories (**Fig. 4**).

In the "standard" multinomial classification problem the category prediction requires to have as many output neurons as categories.

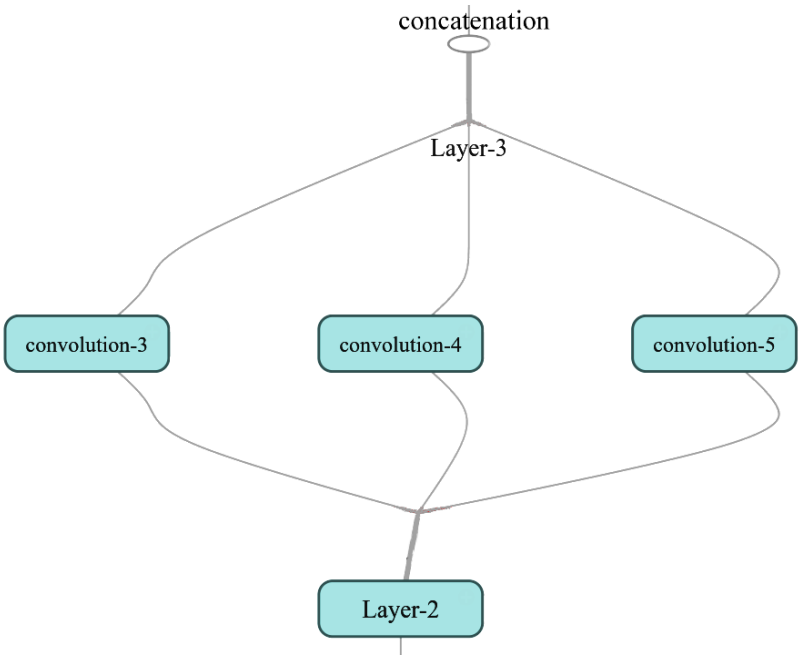

Fig. 3. Architecture of Layer-3 (it consists of three parallel convolutional layers with different window size, input is the output of second layer and produced outcomes are concatenated)

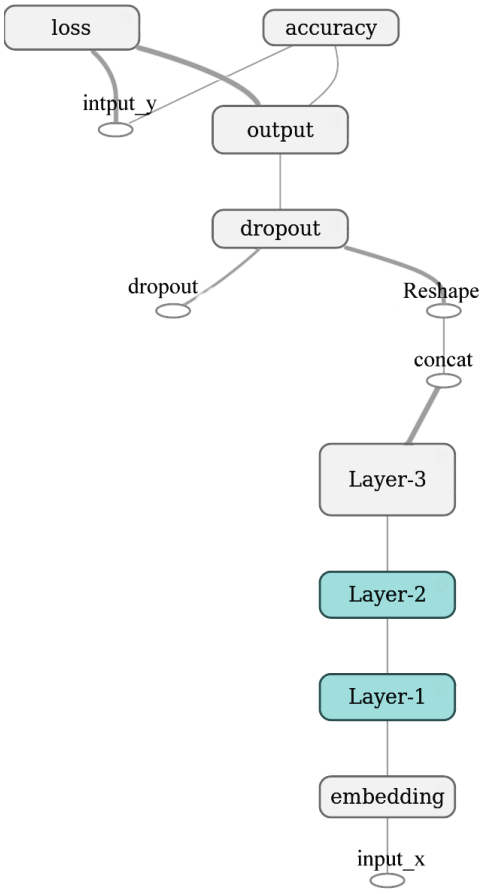

Fig. 4. Graph of the neural network model, describing the whole data workflow (output – fully connected layer, concat – concatenation, input_x and input_y – placeholders for texts and categories)

A neural network is usually trained with a cross-entropy cost function that requires values of the output neurons to represent probabilities – which means that the output "scores" computed by the network for each category have to be normalized, converted into actual probabilities for each category. This normalization step is achieved by means of the softmax function, which is computational expensive when applying for large output layer. That is why the Noise Contrastive Estimation (NCE) [24] was chosen as a lost function. Each training example $(x_{\{1:n\}}, T_i)$ consists of a context, in our case it is text, and a small set of target categories. The following is used as shorthand for the expected count of a category in the set of target categories for a context. In the case of sets with no duplicates, this is the probability of the category given the context:

$$P(y \mid x) = E(Ty) \mid x).$$

The main purpose is to train a function $F(x, y)$ to approximate the log probability of the category given the context.

$$F(x, y) = \log(P(y \mid x)).$$

For each example (x_i, T_i), was pick a set of sampled categories S_i and then a set of candidates was constructed consisting of the sum of target categories and the sampled ones.

$$C_i = T_i + S_i.$$

As a result, the training task is to distinguish the true candidates from the sampled candidates. There are one positive training meta-example for each element of T_i and one negative training meta-example for each element of S_i.

Let's introduce the shorthand $Q(y \mid x)$ to denote the expected count, according to our sampling algorithm, of a particular category in the set of sampled categories.

$$Q(y|x) = E(S(y) \mid x)),$$

where S never contains duplicates,

$$\text{logodds}(y \,\text{came from}\, T \,\text{vs}\, S|x) = \log\left(\frac{P(y|x)}{Q(y|x)}\right) = \log\left(P(y|x) - \log\left(Q(y|x)\right)\right).$$

The first term, $\log(P(y \mid x))$, is what the function $F(x, y)$ would be trained to estimate. In the model there is a layer, which represents $F(x, y)$. The second term, $\log(Q(y \mid x))$, which is computed analytically, is added to it and the result is passed to a logistic regression loss function, whose "label" indicates whether *y* came from T as opposed to S.

$$\text{LogisticRegressionInput} = F(x, y) - \log\,(Q(y \mid x))$$

where $F(x, y)$ is trained by the back propagation signal.

The dropout technique was employed for regularization with a constraint on l2-norms of the weight vectors [25]. The idea behind is to randomly disable neuron that helps to prevent the co-adaptation of hidden units. Let assume $z = [c_1', ..., c_m']$ is convolutional layer, where m corresponds to the number of filters. For computing the output unit dropout uses "masking" vector r. Therefore, in forward propagation, the output unit y was computed as $y = w * z' + b$ where z' is element-wise multiplication of convolutional layer and "masking" vector $z' = z \cdot r$.

Consequently, gradients are backpropagated only through the unmasked units and dropout is used only during the training stage. Additionally, after a gradient descent step the l2-norms constraint is applied simply by rescaling weight vectors, whenever $w_i > s$ set $w_i = s$ [19]. Training is done through stochastic gradient descent over shuffled mini-batches with the Adam update rule [1].

3. Experimental procedures

For convolution neural network model, it is available a lot of hyperparameters for tuning. In order to build our CNN model, an empirical evaluation for hyperparameters was performed and analysis of the effect of hyperparameters on accuracy and productivity was carried out. The model was trained with the following hyperparameters:

- rectified linear units;
- windows (h) of 32, 16, 5, 4, 3;
- dropout rate (p) of 0.5;
- l2 constraint (s) of 5;
- 128 filters per filter size;
- mini-batch size of 35;
- Adam update rule;
- one epoch.

All the experiments were run on Amazon "p2.xlarge" instance with the following hardware characteristics: NVIDIA K80 GPU with 2,496 parallel processing cores and 12GiB of GPU memory, 16GiB of RAM and Intel Xeon E5-2686v4 CPU.

4. Results

Whereas the correct categories were collected together with articles for validation set, the accuracy is calculated simply by comparison the predicted categories with correct ones. Accuracy for each example (text, categories) is defined as the ratio of correctly predicted categories to the total number, sum of predicted and actual, of text categories.

$$\text{Accuracy} = \frac{1}{n}\sum_{k=0}^{n}\frac{C_{i,\text{predicted}} \cap C_{i,\text{actual}}}{C_{i,\text{predicted}} \cup C_{i,\text{actual}}}.$$

The accuracy on the validation set is equal to 84.56±2.6 %. In order to validate our hypothesis about transfer learning articles collected for German language were used. The described above model shows 63.34 +=1.98 % for text written on totally new language, German was not included in our training and testing sets.

5. Discussion

First of all, according to **Fig. 5**, the values of the loss function for train and test dataset, blue and red colors on the figure respectively, start to diverge after 8 thousand samples. This is a strong sign of data overfitting and thinks that increasing the number of words per article will result in better model generalization. Also, it was observed that max-pooling always beat the average pooling and filter sizes depend on concrete task. However, it is found that the L2 norm constraints on the weight vectors and has a little effect on the end result, the similar conclusion was made by Y. Zhang [1].

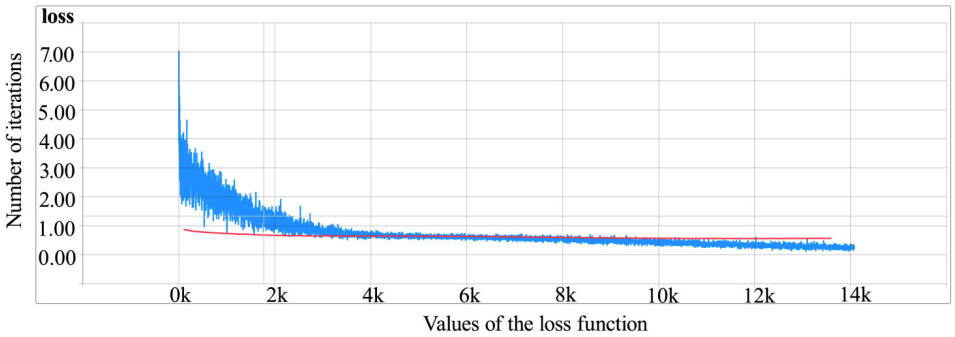

Fig. 5. The values of the loss function (blue for training and red for testing datasets) corresponding to number of iterations

The proposed model has two strong sides: CNNs can be used from scratch without feature extractor and they do not require knowledge of the language, syntax or semantic structures, perform well on new languages and as a result could be used for transfer learning. Even though, there is still a lot of space for architecture research of a model. The accuracy for text in German language show that model effectively applies learned pattern and structure information from other language. It is assumed that approximately 63 % of accuracy is received because of model training on language from the same group (i. e Danish, Norwegian and Swedish).

6. Conclusions

Summing up, CNNs show good results at cross-language classification task and would be a good fit for natural language processing tasks. Despite the fact that convolutional neural networks do not have such clear intuition for location invariance and compositionality as they do in image tasks, it turns out that CNNs perform quite well on NLP problems. The approach in this article works for multilingual long-form texts such as multi-language Wikipedia articles. Additionally, the classification for new language shows promising result but require further research. Intuitively, it makes sense that using pre-trained word embedding would yield larger gains of accuracy for our model, but we used only one-hot words representation and learned such representation during training procedure.

A limitation of this research is that all articles have the same length, as a result, this method may not be a good choice for data that looks considerably different. Although, as a solution, dynamic graph creation approach could be used.

The proposed solution could be used in variety of applications such as search engines, e-libraries, knowledge bases, any information retrieval systems and software that work with multilingual documents.

References

[1] Ko, Y., Seo, J. (2000). Automatic text categorization by unsupervised learning. Proceedings of the 18th Conference on Computational Linguistics, 1, 453–459. doi: 10.3115/990820.990886

[2] Zhang, X., Le Cun, Y. (2016). Text Understanding from Scratch. arXiv:1502.01710v5 [cs.LG]. Available at: https://arxiv.org/pdf/1502.01710.pdf

[3] Korde, V. (2012). Text Classification and Classifiers: A Survey. International Journal of Artificial Intelligence & Applications, 3 (2), 85–99. doi: 10.5121/ijaia.2012.3208

[4] Schäuble, P. (1997). Multimedia Information Retrieval. Springer US, 138. doi: 10.1007/978-1-4615-6163-7

[5] Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems 25 (NIPS 2012), 1097–1105.

[6] Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A. (2015). Going deeper with convolutions. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1–9. doi: 10.1109/cvpr.2015.7298594

[7] Soderland, S. (2001). Building a Machine Learning Based Text Understanding System. Proceedings of IJCAI Workshop on Adaptive Text Extraction and Mining, 64–70.

[8] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J. (2013). Distributed representations of words and phrases and their compositionality. Proceedings of the 26th International Conference on Neural Information Processing Systems, 3111–3119.

[9] Bel, N., Koster, C. H. A., Villegas, M. (2003). Cross-Lingual Text Categorization. Lecture Notes in Computer Science, 126–139. doi: 10.1007/978-3-540-45175-4_13

[10] Oard, D. W. (1998). A Comparative Study of Query and Document Translation for Cross-Language Information Retrieval. Lecture Notes in Computer Science, 472–483. doi: 10.1007/3-540-49478-2_42

[11] Zhou, D., Truran, M., Brailsford, T., Wade, V., Ashman, H. (2012). Translation techniques in cross-language information retrieval. ACM Computing Surveys, 45 (1), 1–44. doi: 10.1145/2379776.2379777

[12] Vulić, I., De Smet, W., Moens, M.-F. (2012). Cross-language information retrieval models based on latent topic models trained with document-aligned comparable corpora. Information Retrieval, 16 (3), 331–368. doi: 10.1007/s10791-012-9200-5

[13] Littman, M. L., Dumais, S. T., Landauer, T. K. (1998). Automatic Cross-Language Information Retrieval Using Latent Semantic Indexing. Cross-Language Information Retrieval, 51–62. doi: 10.1007/978-1-4615-5661-9_5

[14] Li, Y., Shawe-Taylor, J. (2007). Advanced learning algorithms for cross-language patent retrieval and classification. Information Processing & Management, 43 (5), 1183–1199. doi: 10.1016/j.ipm.2006.11.005

[15] Shi, L., Mihalcea, R., Tian, M. (2010). Cross Language Text Classification by Model Translation and Semi-Supervised Learning. Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 1057–1067.

[16] Prettenhofer, P., Stein, B. (2011). Cross-Lingual Adaptation Using Structural Correspondence Learning. ACM Transactions on Intelligent Systems and Technology, 3 (1), 1–22. doi: 10.1145/2036264.2036277

[17] Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C. (2003). A Neural Probabilistic Language Model. Journal of Machine Learning Research, 3, 1137–1155.

[18] Dos Santos, C. N., Gatti, M. (2014). Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics, 69–78.

[19] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). doi: 10.3115/v1/d14-1181

[20] Johnson, R., Zhang, T. (2015). Effective Use of Word Order for Text Categorization with Convolutional Neural Networks. Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. doi: 10.3115/v1/n15-1011

[21] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P. (2011). Natural Language Processing (Almost) from Scratch. Journal of Machine Learning Research, 12, 2493–2537.

[22] Kalchbrenner, N., Grefenstette, E., Blunsom, P. (2014). A Convolutional Neural Network for Modelling Sentences. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). doi: 10.3115/v1/p14-1062

[23] Conneau, A., Schwenk, H., Barrault, L., Lecun, Y. (2016). Very Deep Convolutional Networks for Text Classification. arXiv:1606.01781 [cs.CL]. Available at: https://arxiv.org/abs/1606.01781

[24] Gutmann, M., Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, 297–304.

[25] Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15, 1929–1958.

[26] Zeiler, M. D. (2012). ADADELTA: An Adaptive Learning Rate Method. arXiv:1212.5701 [cs.LG]. Available at: https://arxiv.org/abs/1212.5701